Probabilidad y Estadística

Tarea 1

Integrantes:

Joaquín Barrientos

19.889.690-k

Diego Espinoza

20.072.778-9

Profesor:

Ewald Stark

18 de octubre de 2019

Resumen

En esta tarea, nos dedicamos a trabajar con bases de datos, python y sus librerías respectivas, tales como pandas, que nos proporciona la utilidad DataFrame. Se trabajó con distribuciones de probabilidad, de variables discretas y continuas, por ejemplo, Poisson y distribución normal. Se revisaron varios conceptos, como covarianza, esperanza, desviación estándar.

Para utilizar las bases de datos se utilizó la función DataFrame, que fue de gran utilidad. Son 4 preguntas las cuales se respondieron, pero la pregunta nº2 en específico, se realizó de dos maneras, para datos agrupados y para datos no agrupados.

Pregunta 1

Esta pregunta consistió en resolver el control dos realizado este semestre.

P1.A
Para poder encontrar la probabilidad se utilizó el comando poisson de scipy stats para poder definir la distribución poisson. Luego se utilizó el comando .cdf(x) para poder obtener la sumatoria de probabilidades de "y" menor a "x" y así obtener la probabilidad de 3 visitas en un minuto cualquiera.

P1.B
En esta pregunta se utilizaron los mismos comandos que la PA.1 con la diferencia de que el lambda de poisson es 12 en vez de 4 y el valor de "x" debe ser 5.

P1.C
En este caso se debió ocupar la distribución binomial con n=5 y p=probabilidad obtenida en P1.A. Al igual que en las preguntas anteriores se utilizó scipy stats para definir la distribución binomial y luego utilizar .cdf(2) para obtener la probabilidad de que ocurra como máximo en dos de cinco días.

P1.D
Para resolver esta pregunta se debió utilizar la distribución normal, en donde el enunciado entregaba los valores de la media (que es igual a 5) y la varianza (que es igual a 2). Y nuevamente con el comando .cdf se pudo obtener la probabilidad de que la cantidad de visitas por minutos este entre 4 y 7. Y simplemente se realizó .cdf(7) - .cdf(4).

Pregunta 2

Esta pregunta se realizó para dos casos:

- Datos no agrupados
- Datos no agrupados

Partiremos por el caso de datos no agrupados.

P2.A (datos no agrupados)

Para partir importamos la base de datos con Pandas y su función DataFrame, y luego aplicamos la funcion pd.crosstab para generar la tabla de frecuencias.

P2.B (datos no agrupados)

Como son frecuencias absolutas lo que obtuvimos con las tabla de frecuencias, la mejor opción es hacer un histograma, debido a que este acumula las frecuencias acumuladas para cada valor.

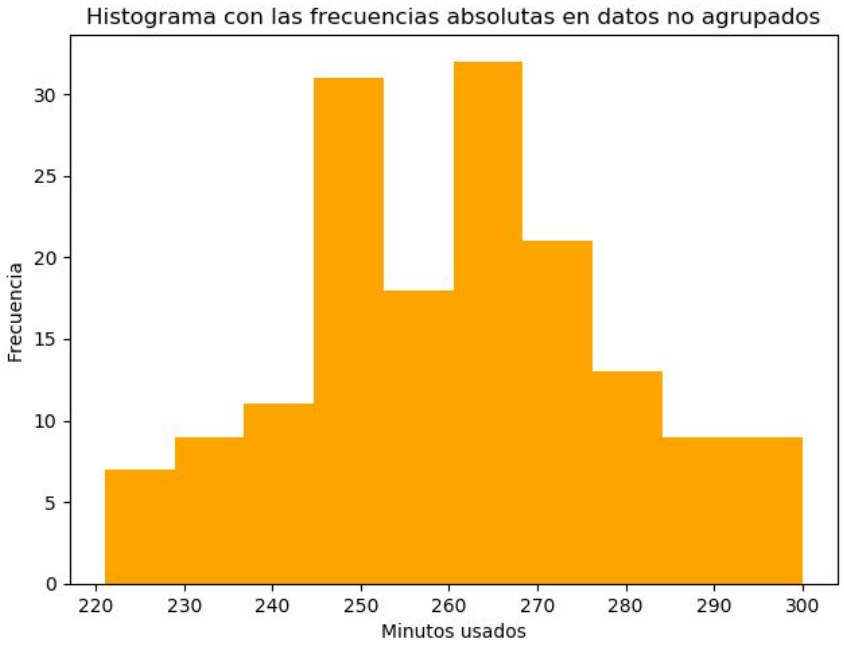


Figura 2.1: Histograma, datos no agrupados.

P2.C (datos no agrupados)

Para esta parte, se debe utilizar la función mean() que sirve para calcular el promedio, por lo que a la base de datos le aplicamos mean() y utilizamos el índice 0.

El resultado del promedio es: 260.64375.

Para calcular la desviación estándar vamos a utilizar la función .std(), la cual se la aplicamos a la base de datos.

El resultado de la desviación estándar es: 17.985789944762143

Con estos datos podemos graficar la distribución normal, la cual se hizo con el siguiente código que fue utilizado en esta pregunta y para la misma pero para datos agrupados, se utilizó una parte de un código encontrado en la página "Python for Undergraduate Engineers". (Kazarinoff. P, 2019)

```
47  d = st.norm(promedio,des_standard)
48
49  x1 = 201
50  x2 = 320
51  z1 = ( x1 - promedio ) / des_standard
52  z2 = ( x2 - promedio) / des_standard
53
54  plt.title("Funcion de Densidad de Probabilidad en datos no agrupados\n Z = (x - μ)/σ , μ = 260.64 , σ = 17.985, x =[221,300] ")
55  plt.ylabel('Probabilidad')
56  plt.xlabel('Valores')
57  x = np.arange(z1, z2, 0.001)
58  y = st.norm.pdf(x,0,1)
59  plt.plot(x,y)
60  plt.legend("Z")
61  plt.fill_between(x,y,y<0, color='blue', alpha=.15)
62  plt.show()
63
```

Figura 2.2: Extracto de código.

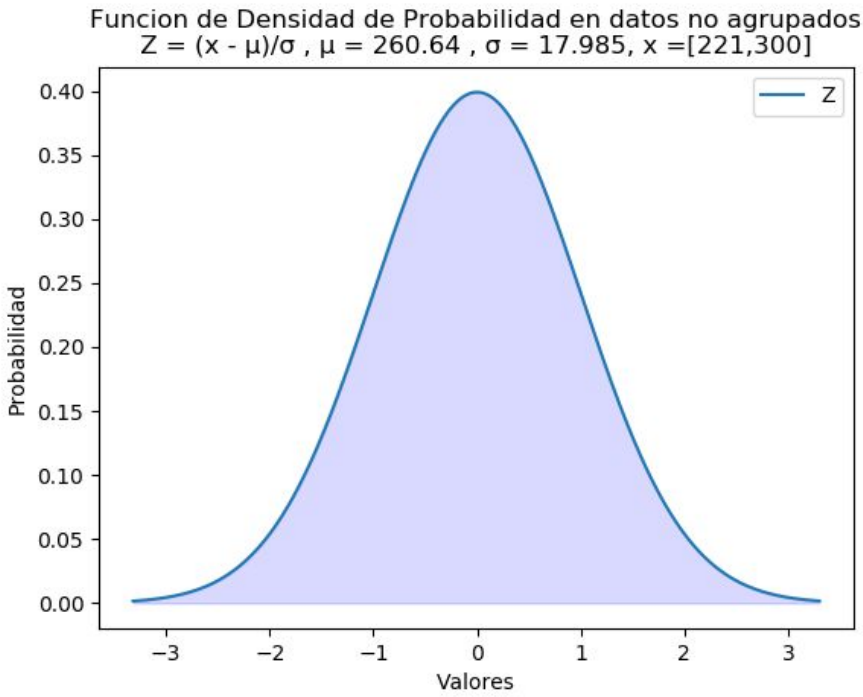


Figura 2.3: Gráfico distribución normal.

Ya con todo esto podemos ver que distribuye de forma normal, y que su promedio y desviación están en los valores esperados.

P2.D (datos no agrupados)

Ya obtenidos los valores del promedio y desviación estándar, podemos sacar el coeficiente de variación. Se calcula de la siguiente forma:

$$CV = \frac{\sigma}{\bar{x}} \quad \text{(ec.1)}$$

El coeficiente de variación es: 0.06900526080046862.
Por lo que un 6,9% de abonados se encuentran dentro de una desviación estándar respecto de la media. Lo que está mucho más abajo de 80% , por lo

que el promedio es representativo del conjunto de datos, por lo tanto el conjunto de datos es homogéneo.

P2.A (datos agrupados)

Para poder crear una tabla de distribución de frecuencias utilizamos un fragmento de un código que sacamos de la página Stack Overflow
(Olascoaga, S).

```
k = 1 + 3.322 * math.log10(len(data_base))
periodos = math.ceil(k)

inf = min(data_base["Numeros"])        # Limite inferior del primer intervalo
dif = max(data_base["Numeros"])
sup = max(data_base["Numeros"]) + 1    # Limite superior del último intervalo

intervals = pd.interval_range(
    start=inf,
    end=sup,
    periods=k,
    name="Intervalo",
    closed="left")

tabla_distribucion_frecuencias = pd.DataFrame(index=intervals)
tabla_distribucion_frecuencias["FreqAbs"] = pd.cut(data_base["Numeros"], bins=tabla_distribucion_frecuencias.index).value_counts()
tabla_distribucion_frecuencias["Marca"]  = tabla_distribucion_frecuencias.index.mid

print(tabla_distribucion_frecuencias)
```

Figura 2.4: Código para distribución de frecuencias.

P2.B (datos agrupados)

Para esta parte, utilizamos las frecuencias absolutas que obtuvimos en la pregunta anterior y con la marca de clase, podemos hacer un gráfico de barras, debido a que el histograma no logró funcionar, pero con plt.bar() es la mejor opción.

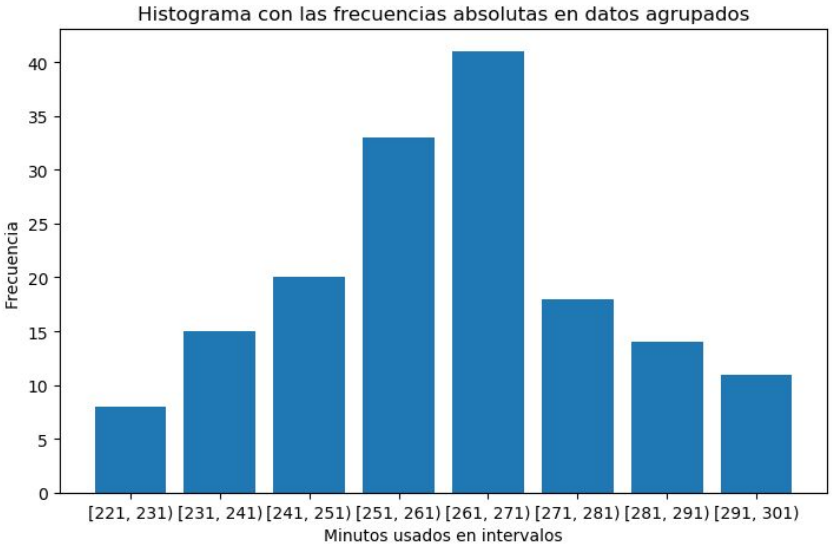


Figura 2.5: Gráfico de barra con marcas de clase y frecuencia absoluta.

Intervalo	FreqAbs	Marca
[221, 231)	8	226.0
[231, 241)	15	236.0
[241, 251)	20	246.0
[251, 261)	33	256.0
[261, 271)	41	266.0
[271, 281)	18	276.0
[281, 291)	14	286.0
[291, 301)	11	296.0

Figura 2.6: Tabla de distribución de frecuencias

P2.C (datos agrupados)

Para obtener el promedio y desviación estándar en datos agrupados es un proceso más complicado el cual fue hecho por nosotros, y el cual se puede ver en la siguiente imagen:

```
116  N = 160
117  X = 0
118  for i in range(len(tabla_distribucion_frecuencias["Marca"])):
119      X += (tabla_distribucion_frecuencias["Marca"].iloc[i])*(tabla_distribucion_frecuencias["FreqAbs"].iloc[i])
120
121  promedio = X/N
122
123  print("El promedio es",promedio)
124
125      #Ahora obtenido el promedio podemos calcular la desviacion estandar.
126  D = 0
127  for i in range(len(tabla_distribucion_frecuencias["Marca"])):
128      D += ((tabla_distribucion_frecuencias["Marca"].iloc[i] - promedio)**2)*(tabla_distribucion_frecuencias["FreqAbs"].iloc[i])
129
130  Desviacion_estandar = math.sqrt(D/(N-1))
131  print("La desviación estándar es",Desviacion_estandar)
```

Figura 2.7: Código para obtener promedio y desviación estándar.

El promedio de la tabla de distribución es: 261.5625
La desviación estándar es: 18.04072524306042
Ya obtenidos estos resultados ya podemos graficar la función de distribución normal.

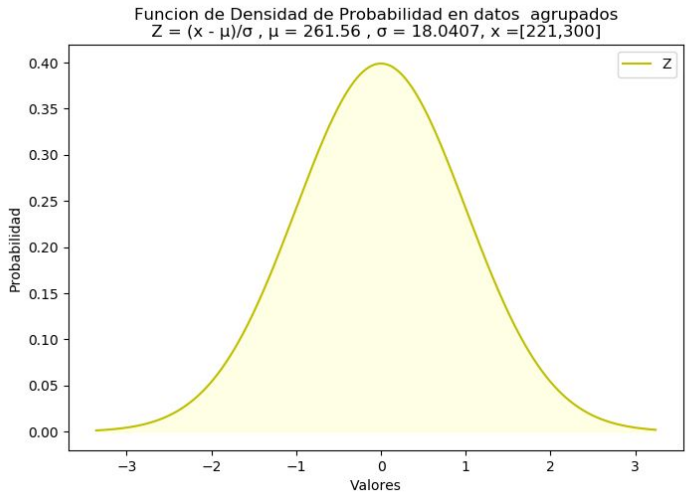


Figura 2.8: Distribución normal para datos agrupados.

P2.D (datos agrupados)

El coeficiente de variación es: 0.06897290415506971
Por lo que un 6,89% de abonados se encuentran dentro de una desviación estándar respecto de la media. Lo que está mucho más abajo de 80% , por lo que el promedio es representativo del conjunto de datos, por lo tanto el conjunto de datos es homogéneo.

Pregunta 3

P3.A

En esta pregunta utilizaremos nuevamente DataFrame.
Podemos inferir que una población se distribuye de una forma normal.
Vamos a seleccionar de la tabla la variables:

- SEMANAS
- PESO
- EDAD_P
- EDAD_M
- TALLA
- HIJ_TOTAL

Las columnas que se eliminaron, fueron en general por dos razones:

1. Debido a que eran cualitativas,pero expresadas en números, por ejemplo, en sexo, eran dos valores numéricos, 1 y 2, correspondiente a hombre y mujer.
2. Tenían datos erróneos, como por ejemplo, cantidad de hijos = 99, lo que "ensuciaba" el análisis

Con la función .drop() eliminamos las columnas no deseadas. Junto con la selección de columnas y la función mean(), sacamos el promedio, que en este caso como distribuye normal, es la esperanza.

Figura 2.9: Vector esperanza, con sus variables.

El vector de esperanzas es el siguiente:
[38.43416094093601, 3302.675426350223, 37.30291624465066, 28.657286505524986, 49.123520662816055, 1.9481034372633288]

Ahora para sacaremos la matriz de varianza covarianza, pero para esto, queremos que todos los datos tengan coherencia entre ellos, por lo que elegimos datos del peso de la guagua, las semanas a las que nació, talla, edad de la madre, y los hijos totales. Creemos que los hijos totales puede ser un factor, o tener un relación con la edad de la madre y otros aspectos.

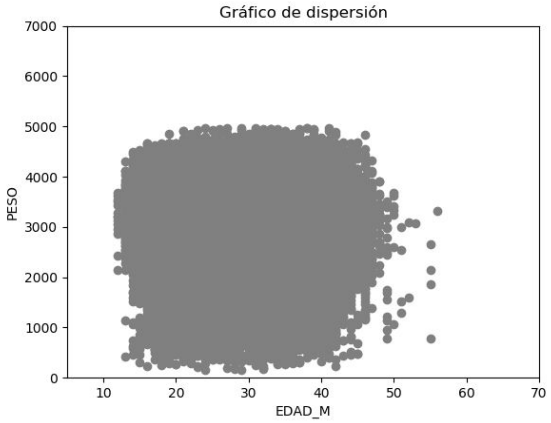
La matriz de covarianza la obtenemos con la función .cov().

P3.B

Esta pregunta se consigue graficando las variables pedidas, con la función plt.scatter().

Es explicativo debido a que a mayor edad es peor el metabolismo en el humano, y puede tener más peso una persona más anciana que una joven por esos motivos. Así el peso del recién nacido también es mayor. También solo los

valores se encuentran entre las edades fértiles de las mujeres.



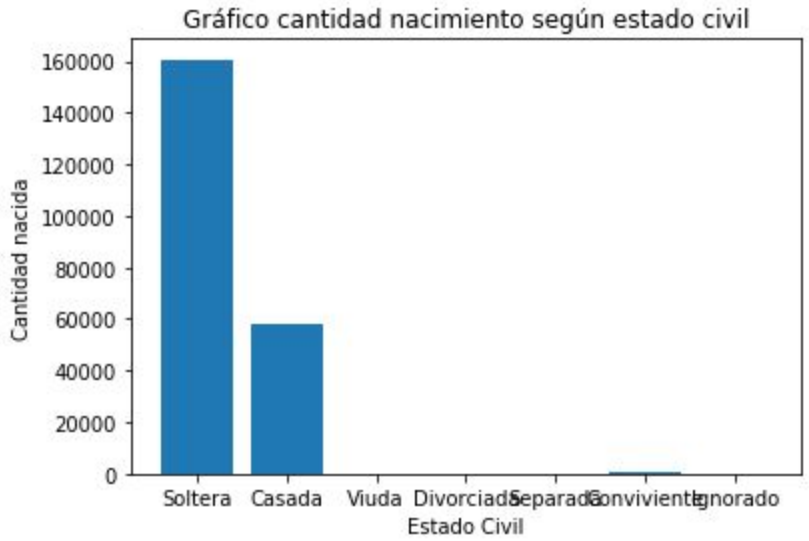
P3.C

Esta pregunta se consigue aplicando la función corr() a la base. Como podemos ver hay variables que tienen una correlación muy alta, como SEMANAS y PESO, lo cual es lógico, debido a que entre más grande el feto, mayor es el peso.
También igual que lo anterior tenemos SEMANAS y TALLA.
Ambas correlaciones son altas y positivas, lo que implica que existe una tendencia lineal positiva entre ambas variables, correspondientemente.
Correlaciones bajas pero positivas son las de PESO y EDAD_M, PESO e HIJ_TOTAL ,SEMANAS e HIJ_TOTAL , TALLA e HIJ_TOTAL, EDAD_P e HIJ_TOTAL, EDAD_M e HIJ_TOTAL lo que implica que no existe una tendencia lineal clara.
Esto parece ser debido a que su relación no es muy significativa, por ejemplo el peso de la guagua y la edad de la mama, no son relevantes entre ellas.
Por otro lado tenemos alguna correlaciones negativas bajas como por ejemplo TALLA Y EDAD M, que no tienen mucha relación.

Pregunta 4

P4.A

Para esta pregunta se utilizó DataFrame. Además consideramos que el gráfico más apropiado para poder ver analizar de mejor manera "cantidad de nacidos según estado civil de la madre" fue el gráfico de barras. Ya que distingue entre cada variable y es muy fácil de analizar visualmente.



Se puede analizar claramente que las mujeres solteras predominan con (al año) 160.728 niños nacidos, segundo lugar se encuentran las casadas con 57748 niños nacidos, luego le sigue las convivientes con 641 niños nacidos, luego las divorciadas con solamente 57 niños nacidos y finalmente las viudas, separadas e ignorados con 0 niños nacidos.

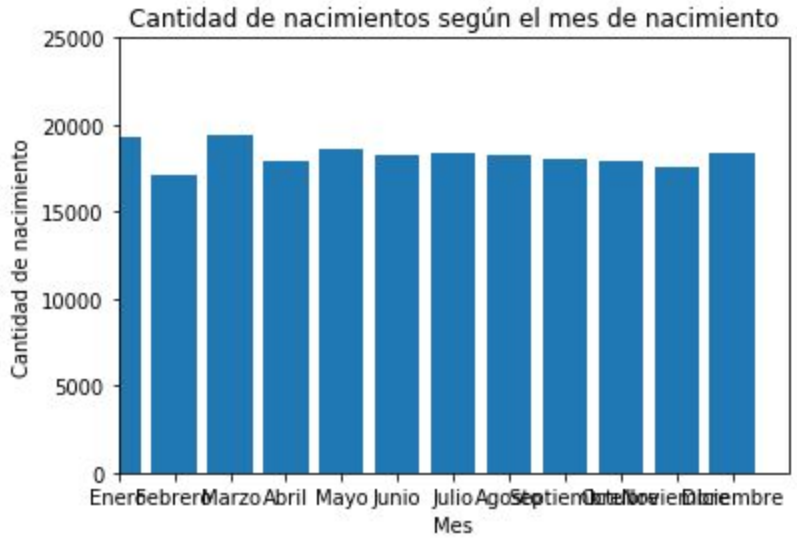
P4.B

En este para poder realizar un histograma en relación al peso de los recién nacidos o se debió inicialmente obtener el tamaño de los intervalos utilizando la fórmula de ni = 1 + 3,334 * log(n), donde n es el tamaño de la muestra. Luego el rango se obtuvo por máximo valor de peso posible menos el menor valor de

peso posible. Luego se determinó los intervalos al realizar la división entre el rango y el ni.

P4.D

En esta pregunta se utilizó nuevamente DataFrame y se realizó un proceso parecido al de P4.A. En donde se contó la cantidad de niños nacidos para cada mes para poder realizar un gráfico de barras.



En el gráfico se pude ver que no existe ninguna tendencia del mes nacido, ningún mes sobresalió y terminó siendo super parejo con una diferencia marginalmente proporcionalmente mínima.

Referencias

Kazarinoff, P. (2019). *Plotting a Gaussian normal curve with Python and Matplotlib*. [online] Python for Undergraduate Engineers. Available at: https://pythonforundergradengineers.com/plotting-normal-curve-with-python.html [Accessed 18 Oct. 2019].

Olascoaga, S. (2019). *¿Cómo hacer una tabla de distribución de frecuencias con python?*. [online] Stack Overflow en español. Available at: https://es.stackoverflow.com/questions/40830/cómo-hacer-una-tabla-de-distribución-de-frecuencias-con-python [Accessed 19 Oct. 2019].